マルチプロセッサ向き目的コードスケジューリングについて (アルゴリズムと計算の理論)

AUTHOR(S):

松原, 義和; 大山口, 通夫; 太田, 義勝

数理解析研究所講究録
1041巻 1998年 241-248

マルチプロセッサ向き目的コードスケジューリングについて

松原 義和　大山口 通夫　太田 義勝
(Yoshikazu Matsubara)　(Michio Oyamaguchi)　(Yoshikatsu Ohta)
三重大学工学部

1 まえがき

スーパスカラ，VLIWのように複数の命令を並列に実行できるプロセッサにおいて，その性能を最大限に発揮するためには，コンパイラによるコード（命令）スケジューリングが重要な役割を果たす[1]．しかしスケジューリング問題は，一般にNP完全である[2]．従って，良い近似解を与える発見的アルゴリズムが求められている．

スケジューリング問題とは，各命令を頂点とし命令間の依存関係を辺で表した有向非循環グラフ（DAG）[3]とプロセッサ数kを与えて，最適な実行終了時間を求める問題である．筆者らは，各命令の親（DAGにおける直接の先行ノード）の数が高々2個である時，線形時間でスケジューリングを行う発見的アルゴリズムを既に提案した[4]．親の数を高々2個としたのは，プログラムの命令として3アドレスコードを扱うとき，レジスタ割り当てを行う前にスケジューリングを実行する場合には，レジスタの再定義による逆依存[1]は存在しないので多くの場合に各命令の親は高々2個に押えられるからである．このアルゴリズムは，クリティカルパスの長さ（DAGにおける頂点の高さ）が$h(\geq 0)$の頂点の集合をS_hと表すとき，S_hとS_{h-1}からなる部分グラフの連結成分に着目してS_hをスケジュールする発見的アルゴリズムであり，各S_hに属する頂点数$|S_h|$が$|S_h| \geq 2k-1$を満たすときには常に最適なスケジューリングを保証する．

本稿では，先ずそのアルゴリズムをクリティカルパス長$h-2$の頂点集合S_{h-2}をも考慮してS_hをスケジュールするものへと拡張することにより，最適性を保証するための制約条件が$|S_h| \geq \frac{3}{2}k$に改善できることを明らかにする．

次に，スケジュール対象のプログラムが配列変数を扱う場合等には，親の数が3個となることがあることを考慮して，本稿では，さらに，[4]で提案されたアルゴリズムを拡張して，親の数が高々3個である時にも有用な，線形時間の発見的アルゴリズムを提案し，各S_hに属する頂点数が，$|S_h| \geq 2k-1$（但し，$k=2$のときは$|S_h| \geq 2k$）を満たすならば常に最適なスケジュールを与えるアルゴリズムであることを示す．

[1]逆依存とは，あるレジスタが再利用されたとき先の命令がそのレジスタを参照した後でなければ後の命令がデータを書き込めないことによる命令間の依存関係である．

2 命令スケジューリング

命令スケジューリングにおいてはプログラムの実行結果を変えないためにデータ依存による制約を考慮しなければならない．データ依存によるスケジューリング順序の制約は，通常DAGを用いて表現する．DAGでは，命令を頂点とし，命令間のデータ依存関係を有向辺で表す．DAGにおいて，命令のクリティカルパス長をその命令から葉への最大パス長と定義する．DAGとプロセッサ数kが与えられたとき，実行サイクル数を最小にする最適スケジューリングを求める問題は，一般にNP完全であることが知られている[2]．

本稿では，次の条件を仮定した発見的スケジューリング・アルゴリズムを提案する．

(1) 各命令は高々n個の親しかもたない（但し，$n=2$又は$n=3$）．

(2) すべての命令が同一サイクルで演算を完了する．

(3) 各プロセッサは均一の機能を持つ．

なお，本稿ではHを最大クリティカルパス長，S_hをクリティカルパス長h $(0 \leq h \leq H)$の命令の集合，kをプロセッサ数とし，$k \geq 2$を仮定する．

3 親の数が高々2個のときのアルゴリズム

3.1 割当てアルゴリズム

本稿で提案するスケジューリング·アルゴリズムは次の割当てアルゴリズムである.

割当てアルゴリズム

(0) 初期化：$l_{H+1} := 0$

(1) $h = H$ とする.

(1-1)

- $l_{h+1} = 0$ のとき　　$S'_h := S_h$.
- $l_{h+1} \neq 0$ のとき

  $k - l_{h+1}$個の空きプロセッサに割当て可能な要素の集合 $V \subseteq S_h$ を任意に選んで割り当てる.　$S'_h := S_h - V$.

(1-2) $S'_h \neq \emptyset$ ならば，以下のように次の時刻以降に S'_hの要素を割り当てる.

$l_h := \mid S'_h \mid \mod k$ を求める．ここで，$\mid S'_h \mid$ は S'_h の要素数とする.

- $l_h = 0$ のとき
  S'_hの要素を任意の順に割り当てる.
- $l_h \neq 0$ のとき
  (a-1) $h = 1$ または $\exists i(h-2 \leq i \leq h), \mid S_i \mid < \frac{3}{2}k$ または $l_h + \mid S_{h-1} \mid \geq 2k$ のとき

  S'_hの要素を「局所割当て方法」(後述) に従って割り当てる．$h := h - 1$;

  (a-2) $\forall i(h-2 \leq i \leq h), \mid S_i \mid \geq \frac{3}{2}k$　かつ $l_h + \mid S_{h-1} \mid < 2k$ のとき

  S'_hと S_{h-1}の要素を「S'_hと S_{h-1}の割当て方法」(後述) に従って割り当てる.
  $h := h - 2$;

$h > 0$ のとき，(1-1) へ戻る.
$h = 0$ のとき，S'_0の要素を任意の順に割り当てる.　□

[「局所割当て方法」による S'_hの割り当て]
頂点集合 $S'_h \cup S_{h-1}$からなる DAG の部分グラフに着目して，その部分グラフの連結成分である $C_1, \cdots, C_m$を，次のΠ_1 と Π_2 に2分割する．但し，S'_hの要素を親，S_{h-1}の要素を子とする．また，有向辺を無向辺とみなして連結成分を得るものとする.

$\Pi_1 = \{C_i \mid C_i$の親の数 $\leq C_i$の子の数$, 1 \leq i \leq m\}$
$\Pi_2 = \{C_i \mid C_i$の親の数 $> C_i$の子の数$, 1 \leq i \leq m\}$

(1) S'_hの要素をΠ_1, Π_2の順に各 C_iごとに割り当てる.

(2) 2つのプロセッサ時刻にまたがって割り当てられる連結成分に対しては，最も多くの子供をもっている要素一つをまず割り当てる．その後，すでに割り当てられた要素と長さ2で連結（一つの子供を介して連結）している要素を順に選ぶ.(連結成分であることにより，これを繰り返すことによって，すべての要素が選ばれる.)　□

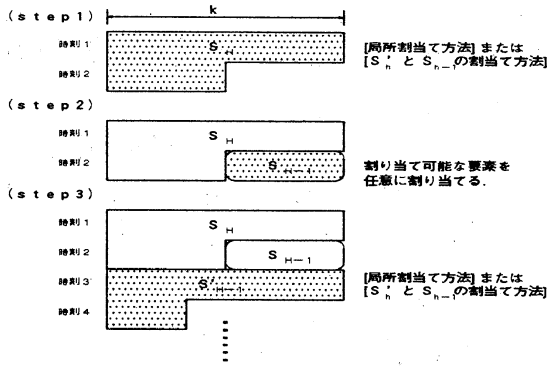

図 1: 割当てアルゴリズム

「S'_h と S_{h-1}の割当て方法」

割当て方法の中では，put, ko, oya は，それぞれ
put$((t_1, \cdots, t_n), X)$：時刻 $(t_1, \cdots, t_n)$ に割当て可能な X の要素すべてを割り当てる.
$X \subseteq S_i$ の とき，
$ko(X) = \{y \in S_{i-1} \mid \exists x \in X.\ x$ は yの先行頂点 $\}$
$oya(X) = \{y \in S_{i+1} \mid \exists x \in X.\ y$は x の先行頂点 $\}$
を意味する．また，S'_hの割当て開始時刻を t とし，S'_hの要素をすべて割り当てたとき，その最終時刻を t'とする $(t' = t + (\mid S'_h \mid - l_h)/k)$.
1．S'_hの要素を出次数の降順に並べたリスト L を作る[2].
2．L の先頭から $2l_h$個を取り出しそれを l_h 個ずつに2分割し，それぞれ $L1, L2$ とする.
$L' = L - \bigcup_{i=1}^{2} Li$,
$Ni = ko(Li)$, $Pi = ko(Ni)$ $(i = 1, 2)$,
$X = N1 \cap N2$ ($\mid X \mid = x$), $A = P1 \cap P2$, $Y =$

[2]出次数が5以上のものは，すべて出次数5とみなしてバケットソートを行う.

$S_{h-1}-(N1\cup N2)$ とする．ここで，一般性を失うことなく，$|N1|\leq|N2|$ とする．
3．$P1\cup P2$ の内から以下の優先順位に従って $k-l_{h-1}$ 個（以後 $k-l_{h-1}=m$ とする）を選択する（その集合を M）．

1 Xの要素のみを親としているもの（その集合を $X1$, $|X1|=x_1$）．

2 親の内の一方を Xの要素とし，他方を $N1\cup N2-X$ の要素とするもの（その集合を $X2$, $|X2|=x_2$）．

3 $N1\cup N2-X$ の要素のみを親としているもの
（その集合を B, $|B|=b$）．
または，親の内の一方を Xの要素とし，他方を Yの要素とするもの（その集合を $X3$，$|X3|=x_3$）．

4 $(P1\cup P2)$ 内から任意に選んだ $2x_1+x_2$ 個
（その集合を C, $|C|=c$）．
（親の内の一方 を $(N1\cup N2-X)$ の要素とし，他方を Yの要素としているもの）　図 2 参照

次の場合分けに従って (3-1) ～ (3-5) のいずれかを行う．

- $|X1\cup X2\cup X3\cup B\cup C|\geq m$　(3-1)
- $|X1\cup X2\cup X3\cup B\cup C|<m$
  - $l_h<\frac{1}{6}k$　(3-2)
  - $l_h\geq\frac{1}{6}k$
    - $|N1|\leq l_{h-1}$
      - $|P1|\leq|S_{h-2}|-m$　(3-3)
      - $|P1|>|S_{h-2}|-m$　(3-4)
    - $|N1|>l_{h-1}$　(3-5)　□

[副割当て方法 (3-1) ～ (3-5)]

- (3-1)
  put$(t, L1\cup L2)$;
  put$(t', oya(M))$; put$(t, oya(oya(M)))$;
  put$((t,\cdots,t'),(S'_h$の残りすべて$))$;
  put$((t', t'+1),(S_{h-1}$の残りすべて$))$

- (3-2)
  put$(t, L1\cup L2)$;
  $S'_h-(L1\cup L2)$ 内から次を満たす $B1, B2, B3, B4, B5$ を選択する．
  $|Bi|=l_h(i=1\sim5)$，$Bi\cup Bj=\emptyset(i\neq j)$
  次に，$ko(ko(Bi))\leq|S_{h-2}|-m$ なる i について
  put(t', Bi) ;
  $S_{h-2}-ko(ko(Bi))$ から m 個を選択する（その集合を M）．
  put$(t', oya(M))$;
  put$((t,\cdots,t'),(S'_h$の残りすべて$))$;
  put$((t', t'+1),(S_{h-1}$の残りすべて$))$

- (3-3)
  put$((t,\cdots,t'-1), L2\cup L')$; put$(t', L1)$;
  put$(t'+1, N1)$;
  $S_{h-2}-ko(N1)$ から任意に m 個を選択する（その集合を M）．
  put$(t', oya(M))$;
  put$((t', t'+1),(S_{h-1}$の残りすべて$))$

- (3-4)
  (3-3) において，$L1$ を $L2$ に，$L2$ を $L1$ に置き換えた手法．

- (3-5)
  put$(t, L1\cup L2)$;
  $Z=\{z\in S_{h-2}\,|\,|oya(\{z\})\cap Y|=1\}$ とする．
  $Y\cup Z$ を頂点集合とする DAG における出次数の大きい方から $\lfloor\frac{m}{2}\rfloor$ 個の Yの要素を取り出す[3]（その集合を Y'）．
  $Z'=\{z'\in Z\,|\,|oya(\{z'\})\cap Y'|=1\}$ とする．
  Z'の要素内から m 個を選択する（その集合を M）．
  このとき，$m-1$ 個しか選択できなかったとき（m が奇数のときのみ）は，$X1\cup X2\cup B$ の内から 1 個の要素を選びこれを加えて m 個とする（その集合を M）．
  put$(t', oya(M))$; put$(t, oya(oya(M)))$;
  put$((t,\cdots,t'),(S'_h$の残りすべて$))$;
  put$((t', t'+1),(S_{h-1}$の残りすべて$))$　□

本割当てアルゴリズムの時間計算量は，$O(n)$ である．（但し，n は総命令数，即ち DAG のノード数である．なお，DAG は条件 (1) を満たしていることから，辺の総数は高々$2n$ で抑えられる．）

3.2　割当てアルゴリズムの性質

補題 3.1 「局所割当て方法」により，時刻 t に S'_hの最後の要素 l $(=|S'_h|\bmod k)$ 個を割り当てたとき，$|S'_h|\geq k+1$　かつ　$|S_{h-1}|\geq k$ であるならば，時刻 t の残り

[3] 出次数が 2 以上のものは，すべて出次数 2 とみなしてバケットソートを行う．

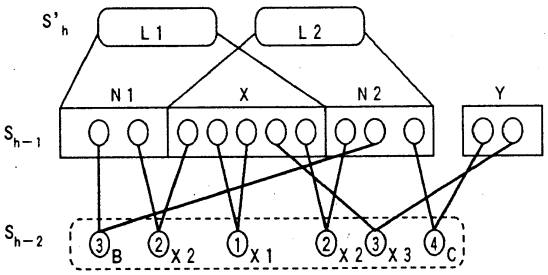

図 2: $X1, X2, X3, B, C$

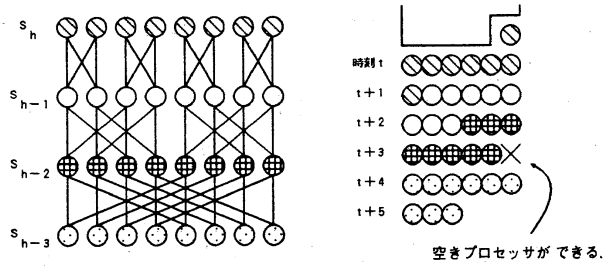

図 3: 空きプロセッサが生じる例（プロセッサ数 $k = 6$, $|S_h| = 8$）

の $k - l$個のプロセッサすべてに S_{h-1} の要素を割り当てることができる.

（証明） 参考文献 [4] , p.971, [補題 2] 参照. □

補題 3.2 $l_h + |S_{h-1}| < 2k$ のとき,「S'_h と S_{h-1}の割当て方法」に従って, S'_h と S_{h-1}の要素を割り当てる.

$$\forall i (h-2 \le i \le h) , |S_i| \ge \frac{3}{2}k$$

ならば, S'_h の割当て最終時刻には S_{h-1}の要素を, S_{h-1} の割当て最終時刻には S_{h-2}の要素を, それぞれ割り当てることができ, 両時刻ともに空きプロセッサは生じない. 但し, $S'_h = S_h - U$, Uは S'_hの要素の割当て開始時刻より 1 時刻前に既に割り当てられている S_hの要素である. また, $l_h = |S'_h| \bmod k$ とする. （証明省略）

定理 3.3 $\forall h (0 \le h \le H)$ について, $|S_h| \ge \frac{3}{2}k$ ならば, 本アルゴリズムは最適スケジューリングを与える.

（証明） $\forall h\ (0 < h \le H)$ について, S_hの割当て開始時刻の次の時刻から（但し, $l_{h+1} = 0$ のときは開始時刻から） 最終時刻まで空きプロセッサが ないことにより最適性を示す.

- S'_hが（割当てアルゴリズムにより）定義された場合
  - (a-1) が適用されたとき
    (i) S'_{h+1}が定義された場合 $l_{h+1} + |S_h| \ge 2k$ であるので, $|S'_h| = |S_h| - (k - l_{h+1}) \ge k$ である.
    (ii) S'_{h+1}が定義されなかった場合 この場合 $|S'_h| \ge k$ である. なぜならば, $|S'_h| < k$ を仮定すると $l_{h+1} + |S_h| < 2k$ であり, また, この場合 $l_{h+2} + |S_{h+1}| < 2k$ なので, S_{h+1} と S_hの要素は 3 時刻以内に割り当てられていることになる. このことは $|S_{h+1}| + |S_h| \ge 3k$ ($\because |S_i| \ge \frac{3}{2}k$ $(i = h, h+1)$) に矛盾する.
    以上より, $|S'_h| \ge k$ である.
    $|S'_h| = k$ のときは明らかに空きプロセッサはない. $|S'_h| > k$ のとき , さらに $|S_{h-1}| \ge \frac{3}{2}k > k$ であるので, 補題 3.1 より 空きプロセッサはない.
  - (a-2) が適用されたとき
    補題 3.2 より空きプロセッサはない.
- S'_hが定義されなかった場合
  即ち, S'_{h+1}が定義され, 場合 (a-2) が適用されたときであり, 補題 3.2 より 空きプロセッサはない.

以上より, 割当て最終時刻以外に空きプロセッサをつくることはない. 故に, 最適スケジューリングである. □

定理 3.3 の条件を満たさないとき, どのような割当てを行なっても, 割当て途中に必ず空きプロセッサが生じる DAG の例を図 3 に示す.

4 親の数が高々 3 個のときのアルゴリズム

4.1 割当てアルゴリズム

次の割当てアルゴリズムにおいては, $S_h \cup S_{h-1}$ からなる DAG の部分グラフを G_hとする.

割当てアルゴリズム

(0) 初期化 : $l_{H+1} := 0$

(1) $h = H, H-1, \ldots, 0$ の順に S_hの要素を割り当てる.

(1-1)

- $l_{h+1}=0$ のとき　$S'_h := S_h$　$U := \emptyset$.
- $l_{h+1}\neq 0$ のとき　$k-l_{h+1}$個の空きプロセッサに割当て可能な要素の集合 $U \subseteq S_h$ を任意に選んで割り当てる．$S'_h := S_h - U$.

(1-2) $S'_h \neq \emptyset$ ならば，次の時刻以降に S'_hの要素を割り当てる．

(a) $h \neq 0$ のとき　$l_h := |S'_h| \bmod k$ を求める．Uの各要素についてその要素を親とするものが少なくとも1つ含まれるように $m(m=2k-1)$ 個の S_{h-1}の要素を選び，それを S'_{h-1}とする．$S_h \cup S'_{h-1}$を頂点集合とする G_hの部分グラフを新たに G_hとする．

- $l_h=0$ または $|S'_h| < k$ のとき　S'_hの要素を任意の順に割り当てる．
- $l_h \neq 0$ かつ $l_h < \frac{1}{2}k$ のとき
  G_hにおける出次数の大きい順に S'_hの要素を割り当てる．
- $l_h \neq 0$ かつ $l_h \geq \frac{1}{2}k$のとき
  G_hにおける出次数が最小である S'_hの要素の1つを xとする．$S''_h := S'_h - \{x\}$．任意に選んだ l_h-1個の S''_hの要素の集合を Vとする．$U \cup V$ の要素を少なくとも一つ親として持つ S'_{h-1}の要素の集合を T_{h-1}とする．
  1. $|T_{h-1}| \geq k$ のとき　先ず，Vの要素を割り当てる．続いて，$T_{h-1} \cup \{s \in (S'_h - V) \mid s$ は T_{h-1}の親 $\}$ を頂点集合とする G_hの部分グラフ G'_hに着目して,「局所割当て方法」(3.1節 参照) に従い，G'_hの親を割り当てる．最後に，まだ割り当てていない $S'_h - V$ の要素を任意に割り当てる．
  2. $|T_{h-1}| < k$のとき　先ず，$S''_h - V$の要素を割り当て，その後 V の要素と x を割り当てる．

(b) $h=0$ のとき　S'_0の要素を任意の順に割り当てる．

□

本割当てアルゴリズムの時間計算量は，$O(n)$ である．(但し，n は総命令数，即ち DAG のノード数である．なお，DAG は条件 (1) を満たしていることから，辺の総数は高々$3n$ で押さえられる.)

4.2　割当てアルゴリズムの性質

定理 4.1 $\forall h(0 \leq h \leq H)$ について，$|S_h| \geq 2k-1$ であり，かつ，割り当て途中に次の　① ② ③ の状態が出現しないならば，本アルゴリズムは最適スケジューリングを与える．ここで，

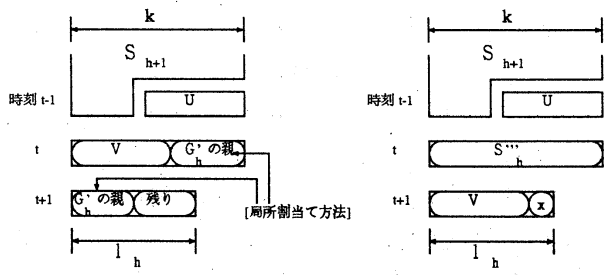

図 4: $l_h \geq \frac{1}{2}k$のときの割当て（ 左：$|T_{h-1}| \geq k$　右：$|T_{h-1}| < k$）

① $k=4$, $l_h=2$　② $k=3$, $l_h=2$　③ $k=2$, $l_h=1$

である．

(証明)　$S_h(0 \leq h \leq H)$ が，割当て最終時刻以外に空きプロセッサをつくることなく割り当てられることにより最適性を示す．

(i) $h=H$ のとき 明らか．

(ii) $h(0<h\leq H)$ のときに，S_hまでが空きプロセッサをつくることなく割り当てられたと仮定して，S_{h-1}の割当てを考える．

- $l_h=0$ のとき 明らか．
- $l_h \neq 0$ かつ $l_h < \frac{1}{2}k$のとき
  [場合 1]　証明後述．
- $l_h \neq 0$ かつ $l_h \geq \frac{1}{2}k$のとき
  1. $|T_{h-1}| \geq k$ のとき　[場合 2]　証明後述．
  2. $|T_{h-1}| < k$のとき　[場合 3（定理 4.1）]
     証明後述．

以上より，割当て最終時刻以外に空きプロセッサをつくらない．従って，最適スケジューリングである．　□

[場合 1 の証明]　S'_hの要素の割当て開始時刻を t，割り当て最終時刻を $t'(>t)$ とする．また，$edge(G_h)$ は G_hの辺の本数を表す．各命令の親の数が高々 3 個であることから，$edge(G_h) \leq 3|S'_{h-1}| = 6k-3$ である．

時刻 t'に割り当てた l_h個の S'_hが支配する子 $(\in S'_{h-1})$ の数が $k+l_h-1$ 以下ならば，$|S'_{h-1}| - (k+l_h-1) = k-l_h$

以上の子供が時刻 t' に割当て可能である．従って，この場合には時刻 t' には S'_{h-1} の要素を空きプロセッサをつくることなく割り当てることができる．

それで，時刻 t' の l_h 個が支配する子の数が $k+l_h$ 以上と仮定する．この時 $l_h < \frac{1}{2}k$ より，$\lceil \frac{k+l_h}{l_h} \rceil \geq 4$ である．従って，時刻 t' 以前（$t-1$ は除く）に割り当てた S_h の各要素は 4 個以上の子を持つ．また，時刻 $t-1$ に割り当てられた $k-l_{h+1}$ 個の S_h の要素が持つ辺の本数は少なくとも $k-l_{h+1}$ であり，S_h のすべての要素が少なくとも辺を 1 本は持つので，S_h のすべての要素が持つ辺の総数は $(\mid S_h \mid -l_h) + (4-1)n \cdot k + (k+l_h)$ 以上となる．但し，$n = (\mid S'_h \mid - l_h) \bmod k$ である．しかしながら，

$$(\mid S_h \mid -l_h) + (4-1)n \cdot k + (k+l_h) \quad \geq \quad 6k-1$$
$$(\because \mid S_h \mid \geq 2k-1\ ,\ n \geq 1)$$

となり，これは $edge(G_h) \leq 6k-3$ に矛盾する．即ち，時刻 t' の l_h 個が支配する子の数が $k+l_h$ 以上となることはない．

従って時刻 t' には S'_{h-1} の要素を空きプロセッサをつくることなく割り当てることができる． □

補題 4.2 D_h と D_{h-1} から構成され，かつ各子供（$\in D_{h-1}$）の親（$\in D_h$）の数が高々 2 個である連結成分 C に対して「局所割当て方法」にしたがって親の割当てを行う．ある時刻に C の親のうちの n 個が割り当てられたならば，次の時刻において少なくとも $n-1$ 個の子供が割当て可能である．

（**証明**） 参考文献 [4]，p.971, [補題 1] 参照． □

[場合 2 の証明] G'_h の親と子の集合をそれぞれ $oya(G'_h), ko(G'_h)$ とする．アルゴリズムにおける G'_h の構成法により，G'_h においては，$ko(G'_h)$ の各要素について，その親の数は高々 2 個であることが保証される．

S'_h の要素の割当て開始時刻を t，割当て最終時刻を $t'(>t)$ とする．

(i) 時刻 $t'-1$ 以前に $oya(G'_h)$ のすべての要素が割り当てられたとき
$\mid T_{h-1} \mid \geq k$ より，時刻 t' には，k 個以上の S'_{h-1} の要素が割当て可能である．

(ii) 時刻 t' に $oya(G'_h)$ の要素が割り当てられたとき
時刻 $t'-1$ と t' にまたがって割り当てられた G'_h の連結成分を C_n とする．C_n の親の数を p_n 個，子供の数を q_n 個とする．$t'-1$ と t' に割り当てられた C_n の親の数をそれぞれ p'_n 個，$p''_n (= p_n - p'_n)$ 個とする．

a. $p_n \leq q_n$ の時
[補題 4.2] より，q_n 個の子供のうち少なくとも $p'_n - 1$ 個は t' において割当て可能である．
また，C_n 以前に割り当てた連結成分 C_i について $p_i \leq q_i$ であるから，t' において割当て可能な S'_{h-1} は，少なくとも ($t'-1$ に割り当てた $oya(G'_h)$ の要素数) $-1 = (k-l_h+1)-1 = k-l_h$ 個は存在する．

b. $p_n > q_n$ の時
[補題 4.2] より，q_n 個の子供のうち少なくとも $p'_n - 1$ 個は t' において割当て可能である．従って，t' において割当て不可能な C_n の子供の数は，

$$\begin{aligned} q_n - (p'_n - 1) \quad &< \quad p_n - (p'_n - 1) \\ &= \quad p''_n + 1 \end{aligned}$$

である．即ち，t' に割当て不可能な C_n の子供の数は，高々 p''_n 個（即ち，t' に割り当てた C_n の親の数）である．
また，C_n 以降に割り当てた連結成分 C_i については，$p_i > q_i$ を満たすから，t' に割当て不可能な C_i の子供の数は高々 p_i である．よって，t' に割り当てられた l_h 個の $oya(G'_h)$ を親の 1 つとして持つ T_{h-1} の要素の数は高々 l_h 個（即ち，t' に割り当てた $oya(G_h)$ の数）である．故に，t' において割り当て可能な S'_{h-1} の要素は，少なくとも $k-l_h$ 個は存在する．

a. b. より，時刻 t' の $k-l_h$ 個のプロセッサすべてに S'_{h-1} の要素を割り当てることができる．

以上より，時刻 t' には空きプロセッサが生じない． □

補題 4.3 [場合 3] の時，$x(\in S'_h)$ が支配する S'_{h-1} の要素数が n であるとき，S'_h の割当て最終時刻に割当て可能な S'_{h-1} の要素は，少なくとも，$\mid S'_{h-1} \mid -(k-1+n)$ 個存在する．

（**証明**） $\mid T_{h-1} \mid < k$ より，$V \cup \{x\}$ が支配する S'_{h-1} の要素が高々 $\mid T_{h-1} \mid + n \leq k-1+n$ 個であることから明らか． □

[場合 3（定理 4.1）の証明] S'_h の要素の割当て開始時刻を t，割当て最終時刻を $t'(>t)$ とする．また，$edge(G_h)$

は G_hの辺の本数を表す．各命令の親の数が高々3個であることから，$edge(G_h) \leq 3 \mid S'_{h-1} \mid = 6k-3$ である．

時刻 t'に割り当てた x が支配する S'_{h-1}の要素数は高々3個である．なぜならば，x が支配する S'_{h-1}の要素数を4個以上と仮定したとき，x は出次数最小の要素であることから，それ以外の $k+l_h-1$ 個以上の S'_hの要素もすべて4本以上の辺を持っていることになる．故に S'_hの要素が持つ辺の総和は，

$$\begin{aligned} 4(k+l_h) &\geq 4k+2k \quad (\because l_h \geq \frac{1}{2}k) \\ &= 6k \end{aligned}$$

となり，$edge(G_h) \leq 6k-3$ に矛盾する．

従って，[補題 4.3] より時刻 t'に割当て可能な S'_{h-1}の要素数は，$\mid S'_{h-1} \mid -(k-1+3) = k-3$ であり，$l_h \geq 3$ ならば，時刻 t'に空きプロセッサは生じない．

問題となるのは，$l_h = 2$, 1 かつ，この [場合3] が適用されるときである．それは，次の3つの場合のみである．

① $k=4$, $l_h=2$　② $k=3$, $l_h=2$　③ $k=2$, $l_h=1$

故に，これらの場合を除けば，S'_{h-1}の要素を時刻 t'に割り当てられるので，空きプロセッサは生じない．□

定理 4.4 割当てアルゴリズムにおいて，*(a)* $(l_h \geq \frac{1}{2}k)$ の場合に $m=2k$とするとき，$\forall h(0 \leq h \leq H)$ について，$\mid S_h \mid \geq 2k$ ならば，本アルゴリズムは最適スケジューリングを与える．

(証明)　[定理 4.1 の証明] における [場合3（定理 4.1）の証明] を [**場合3（定理 4.4）の証明**]（後述）に置き換える．□

[**場合3（定理 4.4）の証明**]　S'_hの要素の割当て開始時刻を t, 割当て最終時刻を $t'(>t)$ とする．また，$edge(G_h)$ は G_hの辺の本数を表す．各命令の親の数が高々3個であることから，$edge(G_h) \leq 3 \mid S'_{h-1} \mid = 6k$ である．

時刻 t'に割り当てた x が支配する S'_{h-1}の要素数は高々3個である．なぜならば，x が支配する S'_{h-1}の要素数を4個以上と仮定したとき，x は出次数最小の要素であることから，それ以外の $k+l_h-1$ 個以上の S'_hの要素もすべて4本以上の辺を持っていることになる．さらに，時刻 $t-1$ に割り当てられた Uの要素が少なくとも $\mid U \mid$ 本の辺を持っている．故に S'_hの要素が持つ辺の総数は，

$$\begin{aligned} 4(k+l_h)+\mid U \mid &\geq 4k+2k+1 \\ (\because l_h \geq \frac{1}{2}k &\quad , \quad \mid U \mid \geq 1) \\ &= 6k+1 \end{aligned}$$

となり，$edge(G_h) \leq 6k$に矛盾する．

従って，[補題 4.3] より時刻 t'に割当て可能な S'_{h-1}の要素数は，$\mid S'_{h-1} \mid -(k-1+3) = k-2$ であり，$l_h \geq 2$ ならば，時刻 t'に空きプロセッサは生じない．

問題となるのは，$l_h = 1$ において，この [**場合3**] が適用される $k=2$ のときのみである．このとき，時刻 t'に割り当てられているのは要素 x 唯一つであり，前記のように，それが支配する S_{h-1}の要素は高々3個である．故に，$\mid S_{h-1} \mid -3 = 1$ 個の S'_{h-1}の要素を時刻 t'に割り当てられるので，空きプロセッサは生じない．□

4.3 定理 4.1　1，2，3 の状況における割当てについて

①

- $\mid S'_h \mid \geq 2k+l_h\ (=10)$ のとき
  S_{h-1}の要素2個を任意に選ぶ（集合 Wとする）．先ず，Wのすべての親を時刻 t から割り当てる．それから，残りの S'_hの要素を割り当てる．

  Wの親の総数は高々6であるので，それらをすべて時刻 t'より前の時刻に割り当てることができる．故に，時刻 t'には Wが割当て可能となり空きプロセッサは生じない．

- $\mid S'_h \mid < 2k+l_h\ (\mid S'_h \mid = 6)$ のとき
  Uの要素を親とする S_{h-1}の要素を任意に一つ選び x とする．

  (i) x が Uに2個以上の親を持つとき　任意に $y \in S_{h-1}-\{x\}$ を選んで $W=\{x,y\}$ とする．

  (ii) x が Uに1個のみ親を持つとき
  $y \in S_{h-1}-\{x\}$ が，(1) Uの要素を親としているか，(2) 親の数が2以下か，(3) x と共通の親を持つ，ならば，$W=\{x,y\}$ とする．(1)(2)(3) のどれも満たさないならば，$S_{h-1}-\{x\}$ から任意に2個の要素を選んで Wとする．

  S'_hの要素を，Wの親から優先的に時刻 t 以降に割り当てる．Wの親（$\subseteq S'_h$）の数は高々4個といえるので，時刻 t'には Wが割当て可能となり空きプロセッサは生じない．

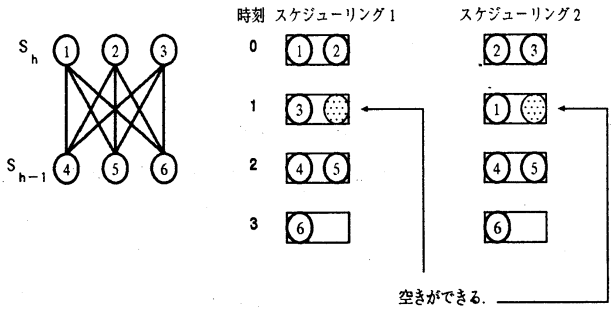

図 5: 空きプロセッサが生じる例（プロセッサ数2個）

② 各要素について親の数は高々3個であることから，S_{h-1}のある要素のすべての親を時刻$t'-1$に割り当てることにより，時刻t'にはそのS_{h-1}の要素が割当て可能となる．

③ この場合には，時刻t'にはS_{h-1}の要素を1個も割り当てることができないDAGが存在する（図5）．即ち時刻t'に空きプロセッサが生じる場合がある．

以上より，①②については，これらの場合に対応したアルゴリズムを付加することにより，空きプロセッサを作ることなく割り当てることが可能である．

なお，本稿で用いた条件（2）（3）に加えて$k=2$を仮定した場合には，常に最適スケジューリングを保証する多項式時間アルゴリズムが文献[5, 6]に与えられている．

5 むすび

本稿では，各命令の親の数が高々2個であるとき，及び，高々3個であるときの2通りの条件のもとで，線形時間でスケジューリングを行う発見的アルゴリズムを提案し，各S_hに属するノード数がある条件を満たすとき最適なスケジューリングを与えることを示した．

参考文献

[1] 小松秀昭，神力哲夫，古関 聰，深澤良彰，“命令レベル並列アーキテクチャのためのレジスタ割付け技法”，情報処理学会論文誌，Vol.36 No .12, pp.2819-2830,Dec. 1995.

[2] J.D.Ullman , “NP-Complete Scheduling Problems”, J.Comput. Syst. Sci. 10 ,pp.384-393 , 1975.

[3] Aho,A.V. , D.Ullman, and R.Sethi, “Compilers Principles, Techniques, and Tools”, Addison-Wesley,Reading,MA, 1986.

[4] 松原義和, 服部忠幸, 大山口通夫, 太田義勝，“並列処理を考慮した目的コードスケジューリング”，電子情報通信学会論文誌 Vol.J80-D-I,No.12 , pp.971-974, 1997.

[5] E.G.Coffman and R.L.Graham, “Optimal scheduling for two-processor systems”, Acta. Inf. 1,pp.200-213, 1972.

[6] M.Fujii, T.Kasami and N.Ninomiya, “Optimal sequence of two equivalent processors”, SIAM J. Appl. Math.17, pp.784-789,1969.